Procedia Computer Science

Volume 80, 2016, Pages 814–823

ICCS 2016. The International Conference on Computational Science

Efficient Strategy for Collective Navigation Control in Swarm Robotics

Luneque Silva Junior[1] and Nadia Nedjah[2]

[1] Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil
luneque@cos.ufrj.br
[2] State University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil
nadia@eng.uerj.br

Abstract

In *swarm robotics*, it is necessary to develop methods and strategies that guide the collective execution of tasks by the robots. The design of such tasks can be done considering it as a collection of simpler behaviors, called *subtasks*. In this paper, the *Wave Swarm* is presented as a general strategy to manage the sequence of subtasks that compose the *collective navigation*, which is an important task in swarm robotics. The proposed strategy is based mainly on the execution of wave algorithms. The swarm is viewed as a distributed system, wherein the communication is achieved by message passing among robot's neighborhood. Message propagation delimits the start and end of each subtask. Simulations are performed to demonstrate that controlled navigation of robot swarms/clusters is achieved with three subtasks, which are recruitment, alignment and movement.

Keywords: Swarm Robotics, Wave Algorithms, Collective Navigation, Task Sequentialization

1 Introduction

A *swarm* of robots is a group of mobile agents working cooperatively to achieve some *collective behavior* [1]. Individual swarm robots are devices with limited capabilities of computational processing, sensing, communication and movement. Swarm robotics approach contrasts with the idea of a single, complex and expensive robot performing tasks that are hard or unhealthy for humans. To perform different kinds of tasks, the robotic swarm must be robust, flexible and scalable, which are characteristics of many multi-agent systems [2]. Such tasks deal with the spatial organization among robots, the motion of the robots in the search of objects and collective decision-making. The tasks performed by robotic swarms have been extensively investigated, as it can be seen in [3, 4, 5, 1, 6, 7]. In this paper, the behavior of an individual robot is defined by a local algorithm that makes use of basic concepts of distributed systems, such as the *direct communication* with explicit messages between processes. Nonetheless, swarm robotics

 Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2016

doi:10.1016/j.procs.2016.05.371

may also use *indirect communication*, sensing the presence of robots or interacting with the environment.

A way to make the swarm able to perform complex tasks is the division of the entire behavior into *subtasks*. Usually, the behavior describes, in a high level of abstraction, the work of the swarm as a whole. This behavior can be interpreted as a sequence of specific subtasks. In this work, a class of distributed algorithms, termed *wave algorithms*, is exploited to mark the start and end of sequential subtasks. Furthermore, the swarm may be divided into groups to perform the same - or different - subtasks in parallel. In *heterogeneous swarms*, it is possible to group robots according to their functionalities. However, in a *homogeneous swarm*, there is no physical difference between the swarm members. In this case, robots must collectively achieve a consensus to form clusters. Wave algorithms can be used to form a group of robots in a subtask called *recruitment*. The characteristics of this wave-based recruitment was been studied in previous works, as it can be seen in [8, 9]. In this approach, the number of groups is defined by the number of *recruiter robots* present in the swarm.

For many tasks performed by of mobile robots, such as the *foraging* [4] and the *collaborative transport* [10], the agents must be able to move in the environment without losing contact with the group members along the way. In this work, this *collective navigation* is achieved with three sequential subtasks: (*i*) the recruitment of robots, forming one or more clusters, (*ii*) the alignment of robots relative to the direction of the recruiter robot of the respective group, and (*iii*) the movement of robots in a leader-following scheme.

The division of the swarm's work into subtasks was explored in the work of Brutschy et al. [11]. In their work, the execution of a task is controlled by multiple hardware devices called *task abstraction modules*. It informs the robots when each subtask starts and ends. In the study of navigation, the *boids*, a flocking control described by Reynolds, is one of the best known in the literature [12]. The flocking is the flight control of multiple agents based on the flock of bands. The agents fly collectively using three rules: alignment, cohesion and separation.

This paper is organized as follows. In Section 2, the strategy of management of tasks based on wave algorithms is presented. The main aspects of the collective navigation addressed in this study are shown in Section 3. Section 4 presents simulation results of different scenarios involving the navigation of single and multiple groups of robots. The conclusion of this paper is presented in Section 5, in which we also point out some relevant future work.

2 Wave Swarm

Numerous tasks in swarm robotics seen in literature are simple behaviors which emerge from the interaction among the robots [13]. Such tasks, even though of hard implementation, are still simple if compared with real-world applications. Such *complex tasks* includes the movement of robots, the identification of characteristics of the environment, the decision-making, and so on. A classical example of complex swarm task is the *foraging*, where the robots need to find and retrieve some object found in the environment [4].

Complex tasks can be achieved in robotic swarms if the implementation considers not only a single and complex behavior, but a sequence of simple or atomic subtasks. With this approach, it is mandatory that the swarm disposes of a management scheme to know when the subtasks begin and when they end. The so-called *Wave Swarm* is a strategy to sequentialize subtasks, enabling the execution of more complex behaviors. It is mainly based on *message passing* among swarm members.

2.1 Network of Connected Robots

A distributed system is considered to be an interconnected collection of autonomous nodes. In general, the nodes can be processes, processors, computers, etc, that have their own private control. Nonetheless, at the same time, a node can be able to exchange information with other nodes [14, 15]. Following this description, a robotic swarm may be considered as a distributed system, with robots being the autonomous nodes and the message passing between then being the node interconnection. This robotic distributed system has an asynchronous timing model, with individual robots processing the steps of a distributed algorithm in an arbitrary order, at arbitrary relative speeds [16].

2.2 Propagation of Information with Feedback

The wave algorithms are a class of distributed algorithms where a starter node, identified as *initiator*, sends messages to its neighbors, which in turn start to send messages to their respective neighbors. A wave algorithm must satisfy the following three requirements:

1. Each computation is finite. In other words, it is a finite sequence of events.
2. Each computation includes at least one decision event.
3. In each computation every decide-event is causally preceeded by an event in each node.

An initiator is a node that starts its local algorithm spontaneously, upon a specific state of an internal trigger. A non-initiator starts its local algorithm upon the arrival of a message, which replicates it to all neighboring nodes except the node from which the message came. The computation of this distributed algorithm is called a *wave*, which is a sequence of events, where each event is a sent/received message. There is also a special kind of internal event called a decide-event. A wave algorithm exchanges a finite number of messages, and then makes a decision. A centralized wave algorithm is that including only one initiator, and all other nodes are non-initiators.

A wave algorithm is used when a message must be spread through a network. An initiator, the node that starts the propagation, must know when all nodes have received the message, then executes a special notify-event. In this case, the algorithm is known as *Propagation of Information with Feedback*, or PIF algorithm [17]. When the propagation message reaches all nodes, feedback messages are sent in the reverse direction. The initiator, thus, knows that the broadcast was correctly done when it receives the feedback from its neighbors. The general structure of the wave algorithm can be seen in Algorithm 1.

2.3 Sequence of Subtasks

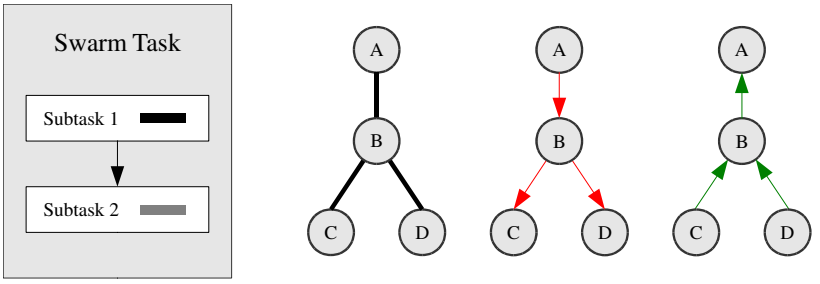
As introduced before, a given *complex task* can be viewed as a sequence of simpler *subtasks*. Figure 1 (a) shows a representation of a swarm task, which can be performed via the sequential execution of two subtasks. In the context of robotic swarms, it is necessary that all robots do the same subtask in a given instant. However, the robots may spend different time executing each subtask. If there is no control in the execution of the task, some robots may start the second subtask, while others are still completing the first one.

To avoid the aforementioned situation, the wave algorithm is used to mark the execution of the subtasks. This creates a dependency during the local execution of the subtask in each robot. To illustrate this, consider Figure 1 (b), which represents the *connectivity graph* of a swarm of four mobile robots. An initiator robot, A, due to physical constraints, can send/receive

Algorithm 1 Wave of Messages

Require: *neighbors_list*;
Ensure: *father_robot*, *child_robots*;
1: **if** initiator **then**
2: broadcast propagation message;
3: receive feedback messages from neighbors;
4: perform the decision event;
5: **else**
6: receive messages;
7: **if** first message **then**
8: define sender as *father_robot*;
9: broadcast propagation message;
10: **else**
11: define sender as *child_robot*;
12: **end if**
13: send feedback message to *father_robot*;
14: wait for the decision of the initiator;
15: **end if**

messages only to/from the robot B. In this case, there is a *father-child* dependency between A and B. This kind of dependency arises as the propagation messages are sent to other robots in the swarm. All non-initiator robots have a *father-robot*. The father of C and D is B, and the father of B is A.



(a) (b)

Figure 1: Sequence of subtasks (a) and a connectivity graph (b).

The precedence diagram in the Figure 2 exemplifies the execution of the two subtasks in the swarm described in the Figure 1 (b). The arrows represents the messages of propagation (arrows down) and feedback (arrows up). The thick lines represent the time spent in the execution of a subtask locally (black for the subtask 1 and gray for 2). A given child-robot can only start a subtask after the reception of a propagation message. On the other hand, the initiator A can only start the next subtask after the reception of the feedback message from B. The dashed lines represent the *waiting time* for the feedback messages (in the robots with children) after the conclusion of its local subtask. The period of time spent by the initiator may look large (in comparison of other robots), but it is necessary to achieve the goal of sequencialization. This strategy is succinct and can be generalized to the case of to numerous tasks.

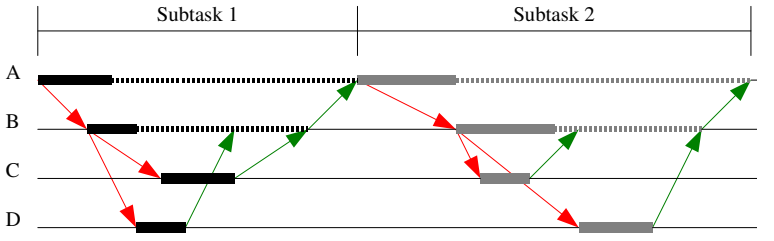


Figure 2: Example of precedence diagram for two subtasks.

3 Collective Navigation

The collective navigation can be defined as the coordinated movement of a group of robots from a point to another of the environment without the loss of swarm members. The movement may be done by all or a well-defined group of robots. In any case, the communication among neighboring robots must be maintained all the way, or else the group will be untied. In other words, the swarm/cluster must move in such way that connectivity among the robots is always guaranteed.

3.1 Robotic Characteristics

A mobile robot is an agent able to move in the environment. In the context of this work, a robot is characterized as having two movement engines, allowing it to go forward and backward, but with no direct right-to-left displacements. Robots with this movement are called non-holonomic. An individual robot is dotted with some computational capabilities. Furthermore, it may sense the environment via some built-in sensors. The main sensory capability allows distance measurement to robots in the swarm. This is a local knowledge, because a robot can only sense robots in a small neighborhood.

Robots may also have some kind of wireless communication, allowing message passing among the swarm members. It may be a local broadcast message, where a robot broadcasts some information to all robots in the neighborhood, or a direct message, identifying both the sender and receiver. This communication must also work as *range and bearing system*. Thus, through the reception of a message, the robot knows the relative distance and direction of its sender.

3.2 Navigation Subtasks

Following the strategy of the wave swarm, the collective navigation can be divided into three sequential subtasks: the recruitment, the alignment and the movement.

The recruitment is the process in which one or more *recruiter robots* build groups inside the swarm [8, 9]. In this subtask, every robot in the swarm knows its *neighborhood*, a list of nearby robots with which communication can be established. The recruiter robot is the initiator of a wave algorithm. When the feedback messages are sent back, these initiators know that the next subtask can be started.

Once the recruitment subtask is completed, the formed groups of robots are not yet ready to move in clusters. At this point, each robot may be heading to a distinct direction. The physical non-holonomic structure of the robots constraints its freedom degree in the combination of two kind of movements: forward/backward displacement, and clockwise/counterclockwise rotation.

To avoid disturbing of the formation, which in turn may lead to disrupting the connectivity among the robots, an alignment subtask is needed, resulting in all robots moving in the same direction. In this approach, the direction of the robots is headed by the direction of the initiator. A wave algorithm is used in the process. The starter robot transmits the relative direction with each neighbor, that iteratively adjusts its position to reach the correct alignment.

To achieve the alignment, the robots must have a bearing system. When a message is received, the robot knows the distance to the neighbor that sent the message, and the relative angle between the direction of that neighbor and the frontal side of the robot. For two robots A and B, the alignment is known when:

$$\Theta_{BA} - \Theta_{AB} = \pi \tag{1}$$

The angle Θ_{AB} is the angle of B relative to A, and Θ_{BA} is the angle of A relative to B.

The initiator of each group starts the alignment wave in the same way it was done during the recruitment. Each non-initiator robot must perform two actions: (i) rotates trying to align itself with respect to the direction of its father-robot and (ii) participates in the alignment of its own child, if any.

All robots, in this subtask, have an internal logical state called *aligned*. The initiator is always in this state. A non-initiator robot is considered aligned when the condition shown in Equation 1 is achieved and its father is already aligned. At this point, the non-initiator stops the rotation, but keeps sending its direction to its children. If it has no child-robot, the robot sends a feedback message to its father. The alignment subtask is completed when the initiator receives feedback messages from all its children.

The coordinated motion of the group of robots uses a *leader-follower* control strategy. The main idea is to keep the robot formation as it was at the start of recruitment. When the wave algorithm is first performed, the resultant spanning tree defines the father-child dependency as well as the relative position between neighbor robots. As the initiator moves, children nodes also start moving, adjusting their positions to maintain the correct distances and angles to the father-robot. There are two rules to be applied during robot motion: the father-following movement and the obstacle avoidance. The initiator has no father to follow, and keeps the forward movement until it meets an obstacle. Any non-initiator robots must keep the correct distance to its father, while avoiding any identified obstacle in its way. An obstacle may be an object in the environment or a neighboring robot that became too close than it should be.

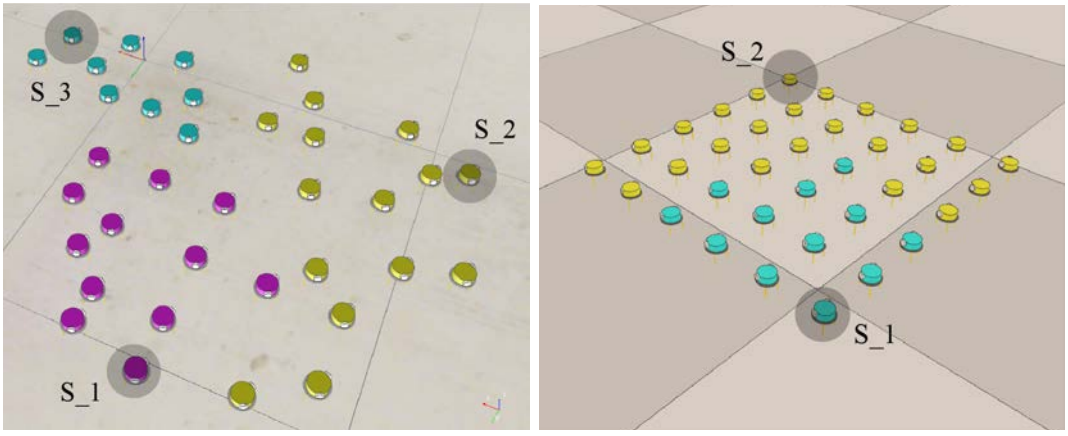
4 Simulation Results

The evaluation of the proposed swarm behavior was done by simulations, using the Virtual Robot Experimentation Platform, V-Rep [18]. It is a commercial 3D simulator with free license for educational use. The V-REP is based on a distributed control architecture: each object/-model can be individually controlled via an embedded script, a plugin, a ROS node, a remote API client, or a custom solution. This makes V-REP very versatile and ideal for multi-robot applications. A simulation model based on Kilobot robot [19] is used. This robot has 3 legs that supports the circuit board above the surface. It moves through vibration resulted by the activation of the two step motors in its circuit board. The vibration allows the robot to move forward and/or make curves. It also has an infrared communication system that allows the robot to send/receive messages to/from neighboring robots. An RGB LED placed in the circuit board can be used to visually signalize specific internal states of the robot. Besides all these characteristics, the Kilobot was chosen to be used in this work because of the similarities

between the programming of the physical robot and the simulated robot model provided by V-Rep. The message spreading of recruitment can be implemented in real Kilobot subject to few minor changes. The actual implementation of coordinated navigation, in other hand, cannot be performed in physical Kilobot, due to the lack of range and bearing system. Real Kilobots can only sense the distance between each other, but not the direction of their neighbors. This restriction was overcome during the simulations, since the robot model can be customized in the V-Rep environment. In the simulated model of Kilobot presented in this work, a distance sensor was included. It can sense the horizontal and vertical distances to other robots in the neighborhood during message reception.

4.1 Recruitment and Alignment

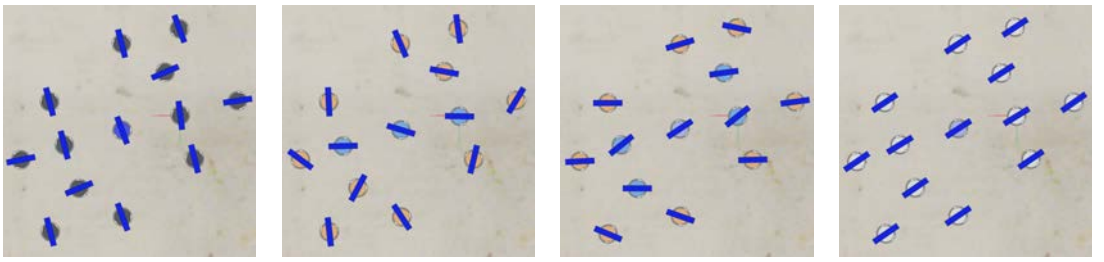
Figure 3 shows snapshots of the recruitment simulations in V-Rep. The result of the formation of three groups in a swarm composed by 36 robots disposed randomly is shown in Figure 3 (a). Three initiators, marked as S_1, S_2 and S_3, start their wave from different regions of the swarm. When the feedback messages start to be sent by child to father nodes, the robot changes its state to the color associated with the initiator. By the end of the execution of the wave algorithm, the swarm is divided into three regions, each one with robots recruited by a initiator. In Figure 3 (b), each swarm member is positioned in a grid of 6×6 robots. Two robots are initiators, marked as S_1 and S_2.



(a) (b)

Figure 3: Recruitment of robots in irregular (a) and regular (b) formations.

In Figure 4 the alignment subtask is verified in a swarm of 13 robots. A mark above each robot indicates its final direction. The initiator robot, in the center, propagates the wave to the neighbors, starting the process of alignment. The other robots change its direction based on the father-child dependence: each one tries to track the direction of its own father robot. When all robots achieve the same direction, feedback messages are sent to initiator, and the subtask ends.

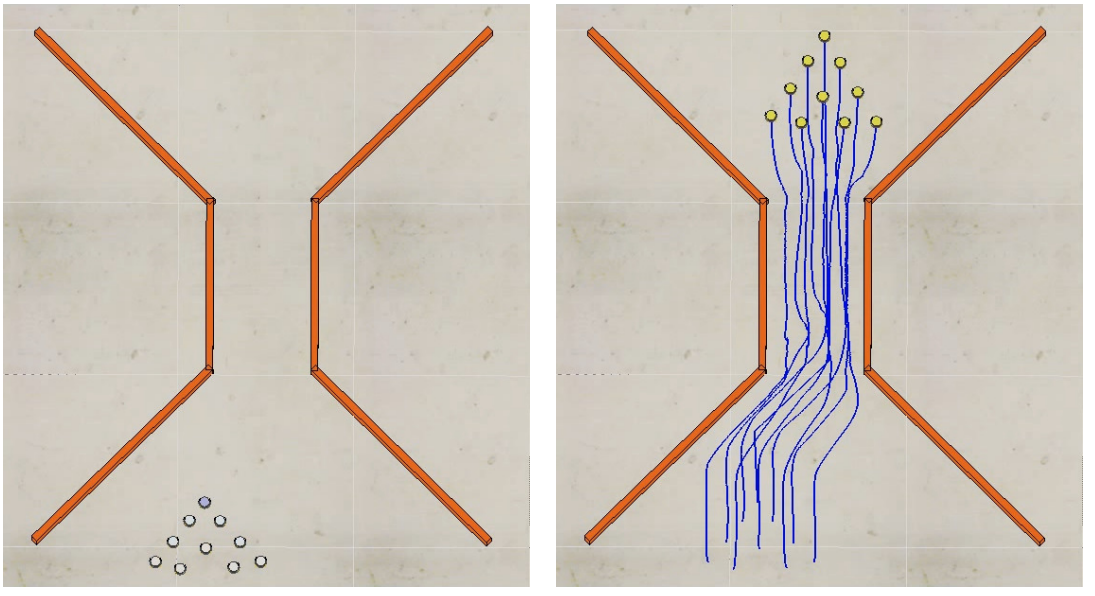


(a) (b) (c) (d)

Figure 4: Robots aligned with the direction of the central initiator.

4.2 Collective Movement

The next step of the navigation is the movement itself. Tests are performed to show the behavior of robots during the displacement in different scenarios. Figure 5 presents a single group of 10 robots in a collective forward movement. The narrowing of the path makes the robots closer to each other, but the adopted policy prevents collisions of robots within the cluster. After crossing this region, the robots adjust their relative positions, restoring the formation as configured at the the start of the recruitment.

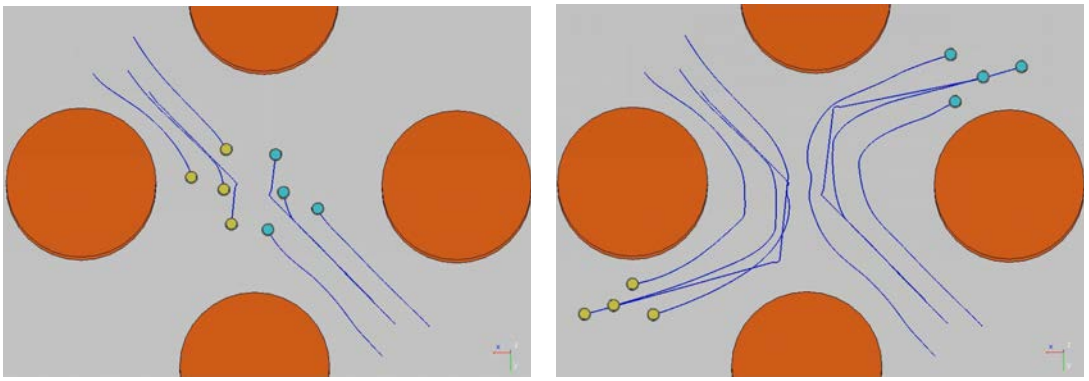


(a) (b)

Figure 5: Forward movement of a group of robots.

It is noteworthy to point out that this movement strategy can be generalized to the case of multiple groups. In Figure 6 two groups of robots are navigating in a collision route. When the robots of one group are close to those in the other group, there occurs a mutual repulsion, causing both groups to change their directions oppositely. After increasing the distance between

the cluster members, each group follows its new route.



(a) (b)

Figure 6: The two initiators avoid the collision changing the direction of the entire group.

In all movement simulations, the formation at the end is almost the same as that at the start of recruitment. Eventually, there is a cumulative error in the position of robots as they move. It is mainly caused by the changes of direction of the initiator in the presence of obstacles. Not being a strictly fixed pattern, the robots have the flexibility to adapt their formation while moving. However, to return to a pattern that is the closest to the initial one, the group must go straight forward, allowing the robots to re-organize their positions. In other words, the initial pattern is restored when the group navigates for some time without running onto obstacles. The movement subtask is very useful, since the formation is not completely know or stored internally by the robots. In fact, the pattern information is distributed among the robots, each one knows just its position in the neighborhood.

5 Conclusions

In this paper, the Wave Swarm is introduced as a general strategy to control the execution of tasks by swarm robotic systems. The robots are organized into a connected topology, with one robot being able to communicate with any other robot within the vicinity in the swarm via a message passing scheme. With this idea, the collective navigation task is divided into three sequential subtasks. Collective navigation in robotic systems is of great interest because it is the basis for many other applications involving the movement of robots. The tests performed in the V-Rep Simulator demonstrate the effectiveness of the method. The navigation is achieved in a sequence composed by the recruitment, alignment and movement subtasks. The swarm could navigate collectively in scenarios with obstacles and in presence of two and four clusters of robots. The tests also show the capacity of each group to restore the relative position of the robots, resulting in the formation preservation. This methodology of design of tasks in swarm systems proved to be very useful. In future works we intend to use this concept in many other tasks, to prove the universality of the Wave Swarm. It includes the movement with multiple groups and tasks related to spatial organization, such as the aggregation/deployment and pattern formations.

References

[1] Erol Şahin. Swarm robotics: From sources of inspiration to domains of application. In *Proceedings of the 2004 International Conference on Swarm Robotics*, SAB'04, pages 10–20, Berlin, Heidelberg, 2005. Springer-Verlag.

[2] Scott Camazine, Nigel R. Franks, James Sneyd, Eric Bonabeau, Jean-Louis Deneubourg, and Guy Theraula. *Self-Organization in Biological Systems*. Princeton University Press, Princeton, NJ, USA, 2001.

[3] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. A taxonomy for swarm robots. In *Intelligent Robots and Systems '93, IROS '93. Proceedings of the 1993 IEEE/RSJ International Conference on*, volume 1, pages 441–447 vol.1, Jul 1993.

[4] Y Uny Cao, Alex S Fukunaga, and Andrew Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4(1):7–27, 1997.

[5] Luca Iocchi, Daniele Nardi, and Massimiliano Salerno. Reactivity and deliberation: A survey on multi-robot systems. In *Balancing Reactivity and Social Deliberation in Multi-Agent Systems*, volume 2103 of *Lecture Notes in Computer Science*, pages 9–32. Springer Berlin Heidelberg, 2001.

[6] Iñaki Navarro and Fernando Matía. An introduction to swarm robotics. *ISRN Robotics*, 2013:1–10, 2013.

[7] Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013.

[8] Luneque Silva Junior and Nadia Nedjah. Wave algorithm for recruitment in swarm robotics. In *Computational Science and Its Applications – ICCSA 2015*, volume 9156 of *Lecture Notes in Computer Science*, pages 3–13. Springer International Publishing, 2015.

[9] Luneque Silva Junior and Nadia Nedjah. (*in press*) Distributed strategy for robots recruitment in swarm-based systems. *International Journal of Bio-Inspired Computation*, 2015.

[10] C.Ronald Kube and Eric Bonabeau. Cooperative transport by ants and robots. *Robotics and Autonomous Systems*, 30(1):85 – 101, 2000.

[11] Arne Brutschy, Lorenzo Garattoni, Manuele Brambilla, Gianpiero Francesca, Giovanni Pini, Marco Dorigo, and Mauro Birattari. The tam: abstracting complex tasks in swarm robotics research. *Swarm Intelligence*, 9(1):1–22, 2015.

[12] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, pages 25–34, New York, NY, USA, 1987. ACM.

[13] Levent Bayındır. A review of swarm robotics tasks. *Neurocomputing*, 172:292–321, 2016.

[14] Valmir C. Barbosa. *An Introduction to Distributed Algorithms*. MIT Press, Cambridge, MA, USA, 1996.

[15] Gerard Tel. *Introduction to Distributed Algorithms*. Cambridge University Press, New York, NY, USA, 2nd edition, 2001.

[16] Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.

[17] A. Segall. Distributed network protocols. *IEEE Transactions on Information Theory*, 29(1):23–35, 1983.

[18] Eric Rohmer, Surya PN Singh, and Marc Freese. V-rep: A versatile and scalable robot simulation framework. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1321–1326. IEEE, Nov 2013.

[19] Michael Rubenstein, Christian Ahler, Nick Hoff, Adrian Cabrera, and Radhika Nagpal. Kilobot: A low cost robot with scalable operations designed for collective behaviors. *Robotics and Autonomous Systems*, 62(7):966 – 975, 2014.